CS/SE 3RA3: Sample Solutions to Assignments 2

Total of this assignment is 105 pts. Each assignment is worth 10%.

Note: If you think your solution has been marked wrongly, write a short memo stating where marking in wrong and what you think is right, and resubmit to me during class, office hours, or just slip under the door to my office.

Q1 [10]

Both the IEEE Std-830 requirements document and the Volere requirements document are on a separate file.

Both documents had a lot in common, a lot of the content was the same and both outline the main functional and non-functional requirements. Though formatted differently, all the requirements and constraints were listed in both documents. Volere however, went into further detail and required us to describe the scope of work, risks involved in the project, along with the various project issues at the bottom of the document.

For this project I believe that the Volere template is a better pattern to follow for the CyberFridge system. As it is a brand new system never developed before, further detail is necessary as we need to ensure all aspects of requirements, risks and constraints are addressed. The IEEE lacks some information that the Volere requires us to provide and though it is clear and concise, discrepancies on the information in the SRS is not ideal. By covering the extra topics such as risks and project issues, developers are able to make sure all problems are addressed and the system is implemented properly.

Q2 [4=2+2]

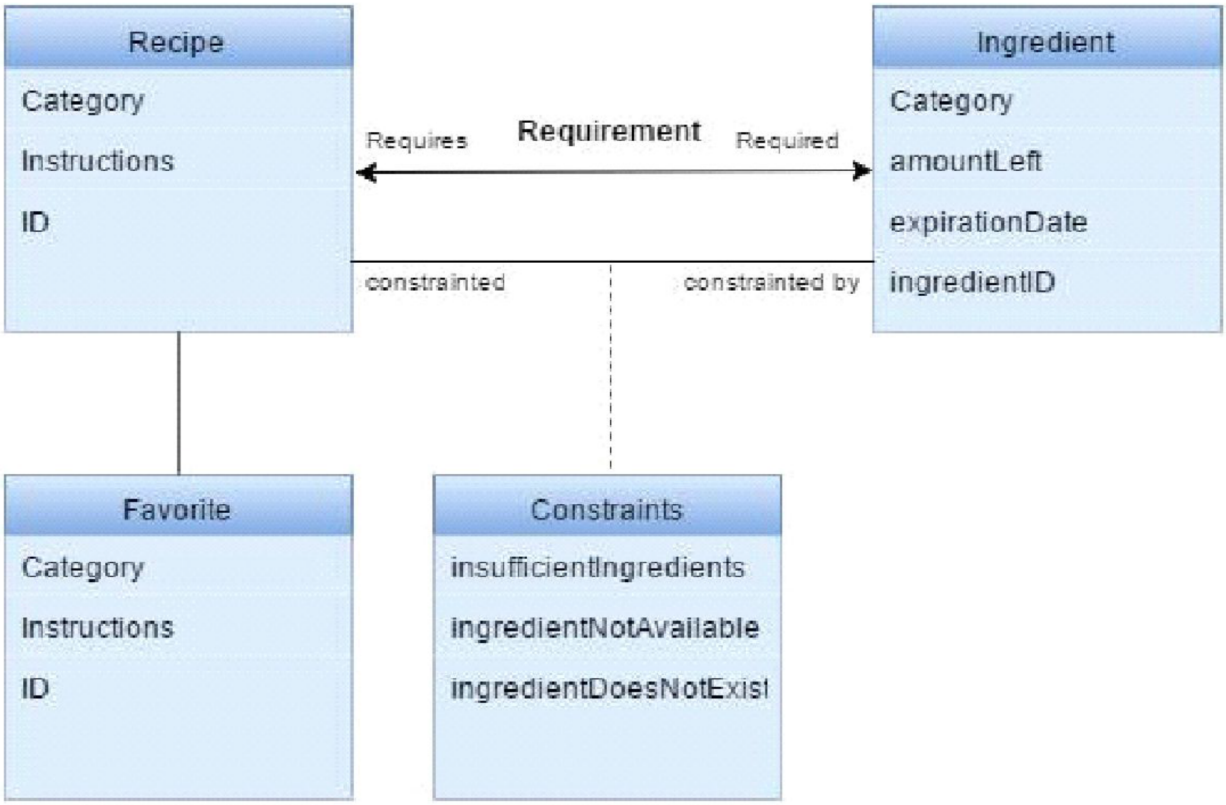
a) Maintainability and Portability Requirements
Portability Requirements – CyberFridge is expected to be able to communicate with Windows, Apple, or Linux devices.
Fit Criteria: The CyberFridge is expected to be able to communicate with at least 90% of Windows, Apple, or Linux devices, older models with those OS may not be able to handle the new system.
b) Security Requirements
Confidentiality – A password is required to access personal information such as recipe interests, and to view any information remotely.
Fit Criteria: The user's password must be a "strong" password to ensure/lower the risk of compromising information. A "strong" password must have a minimum length of 8, contain a lower case and upper case letter, a number and a symbol.

Q3 [10]

Build an entity-relationship diagram to declare and structure the data of the systems.

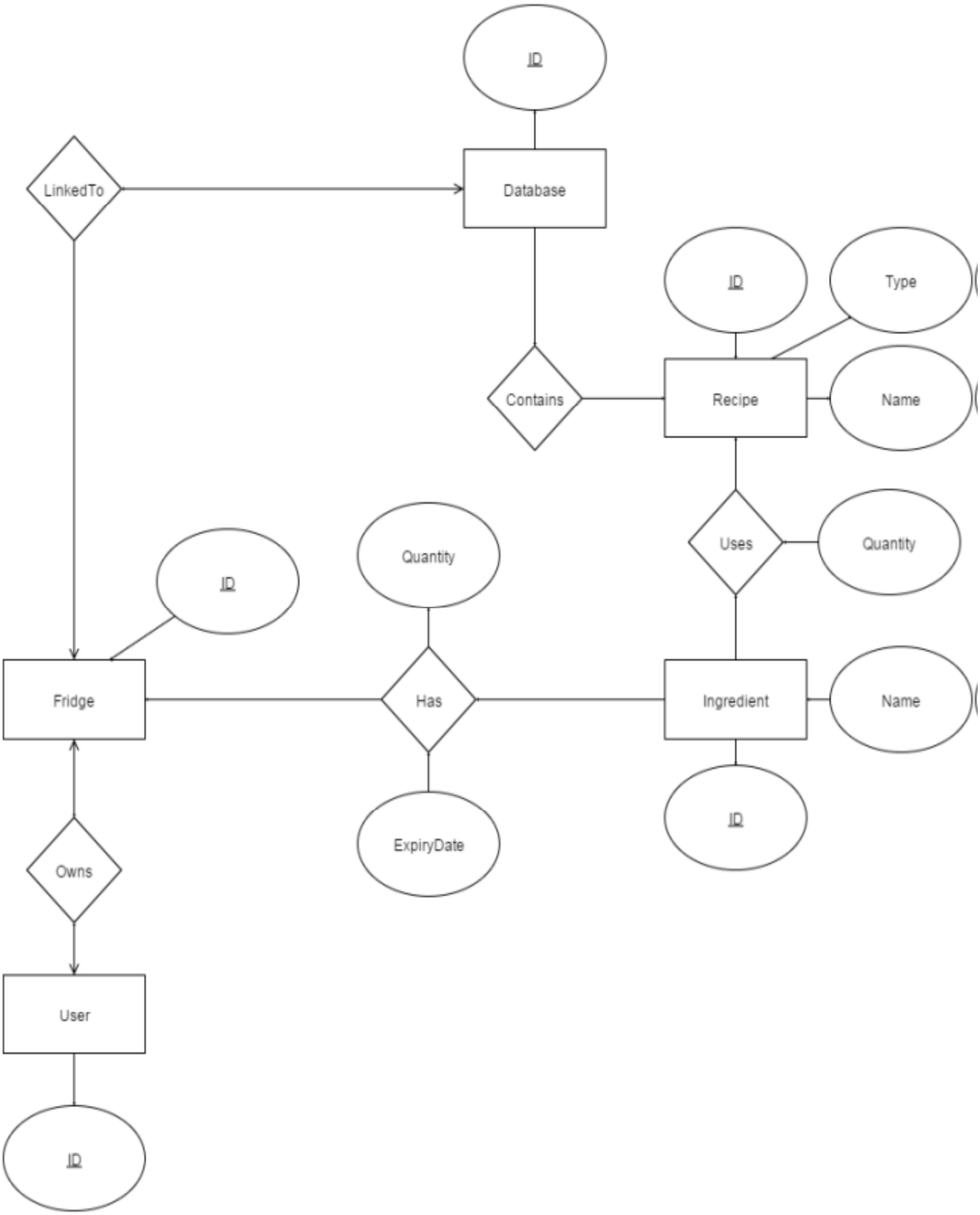
Solution1: UML Based

Entity-Relationship Diagram is UML based version of the entity-relationship diagram. It is easier to understand and follow as opposed to the one used in databases. This version does not require you to understand what each shape and symbol means. Also, the UML based version fully labels the diagram and in my opinion, provides greater detail.



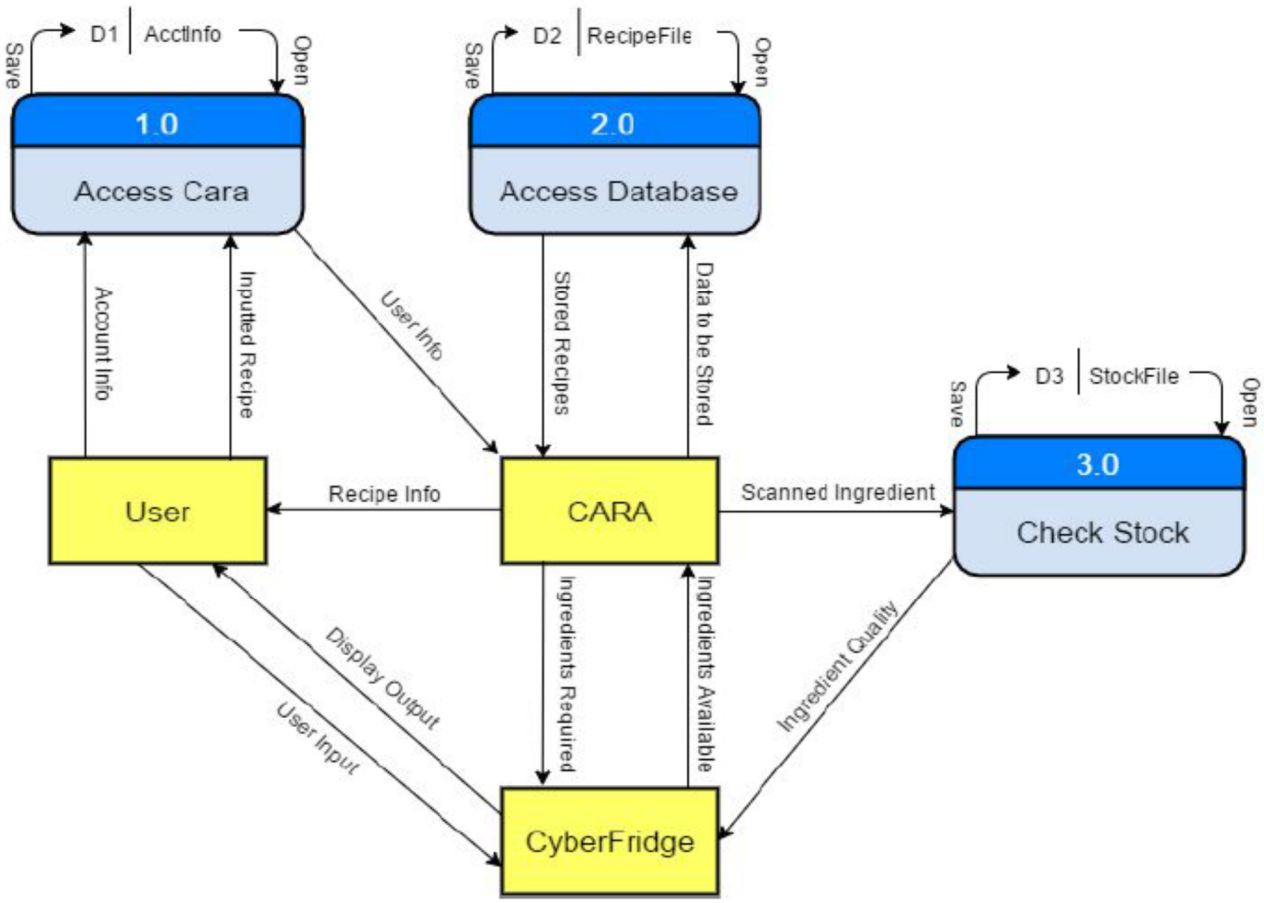
Solution2: Version used in databases

The structure used is the version used in databases. It is easier to read as entities, relationships, and attributes have different shapes to identify them rather than being crammed together. Also, where the database version is better is in identifying relationships. Relationships are represented by a distinct diamond that connects entity sets rather than just a word on a line, the diamond is much clearer to see that it is a relationship. It is also easier to see the distinction between an entity set and its attributes in the database version.


ID
LinkedTo
Database
ID
Type
Contains
Recipe
Name
Uses
Quantity
Quantity
ID
Fridge
Has
Ingredient
Name
ID
ExpiryDate
Owns
User
ID

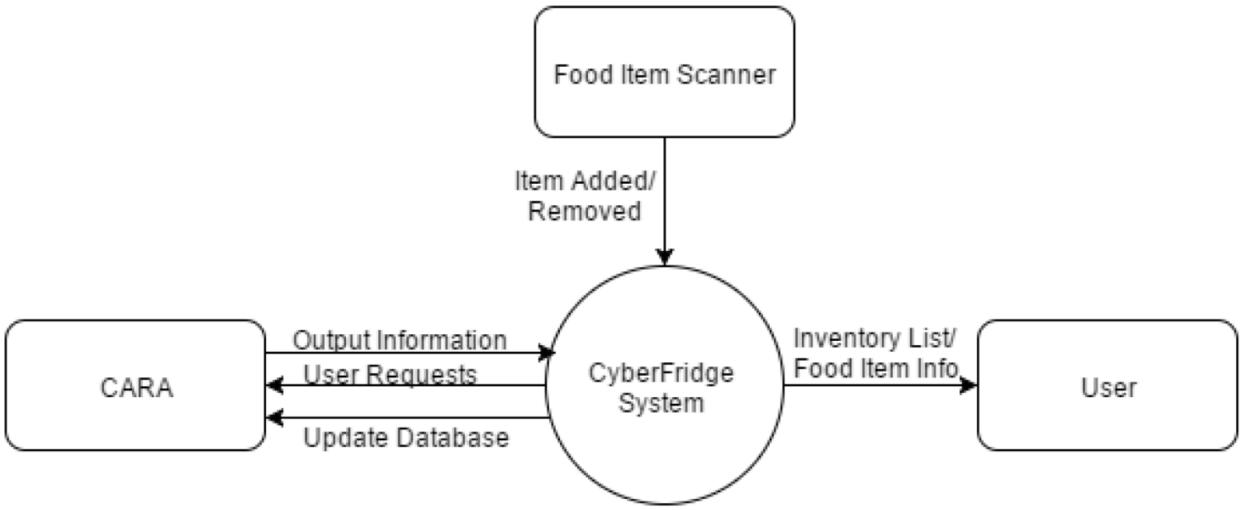
Q4 [10])

Data-Flow Diagram



Q5 [10]

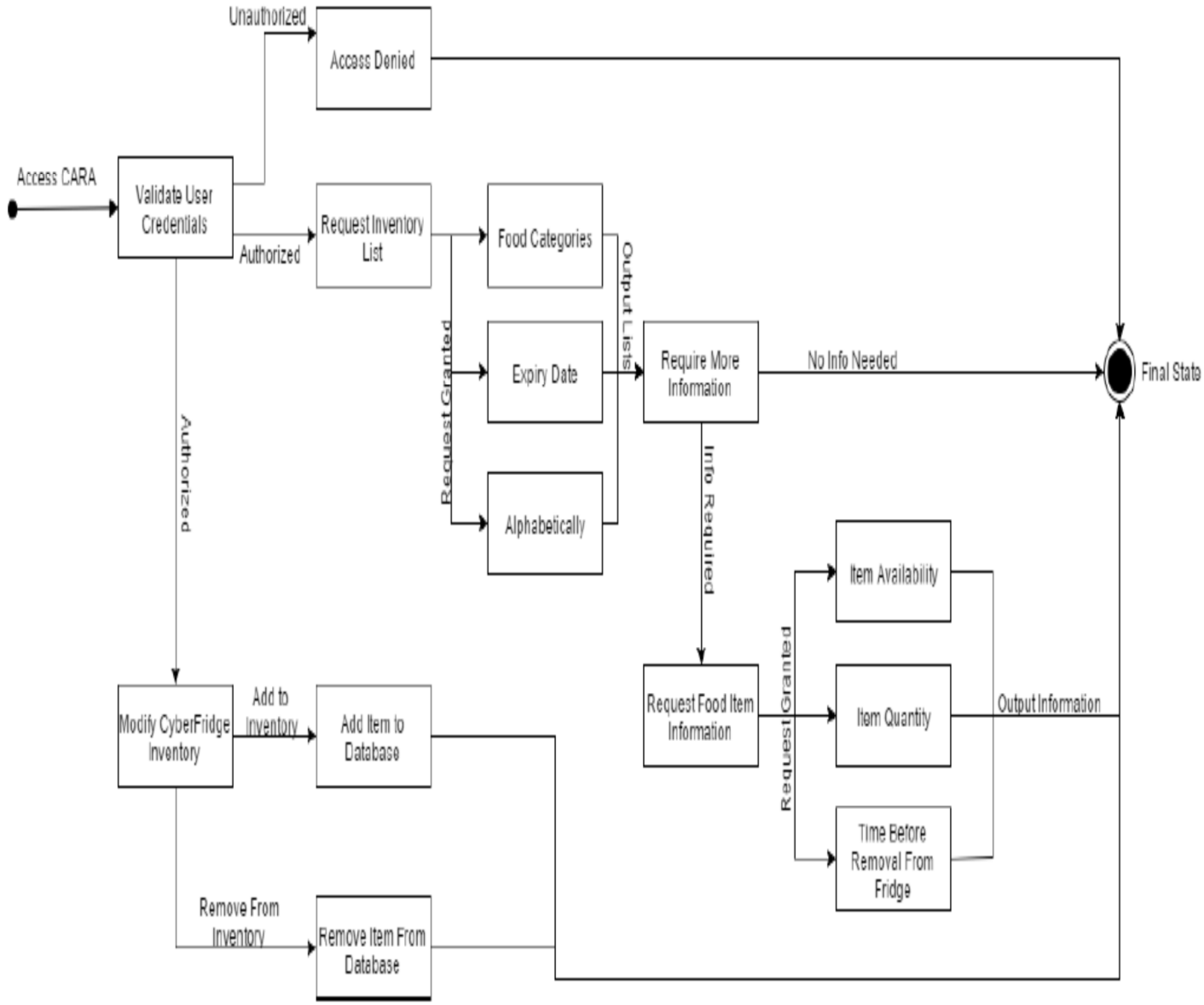
Context Diagram:



Use Case Diagram:

Q6 [10]

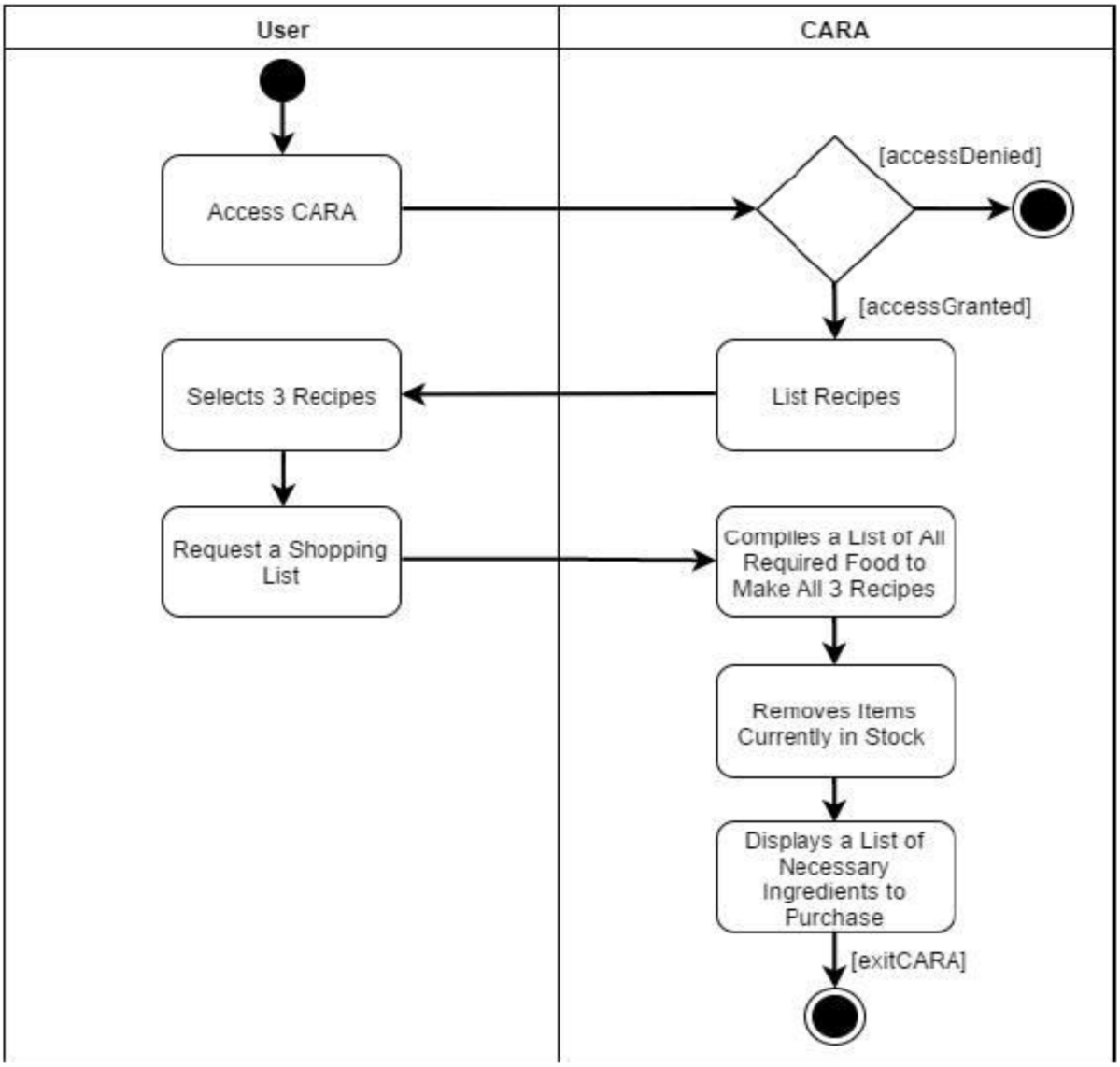
State Machine Diagram to keep track of food items currently in the kitchen and information about each item.



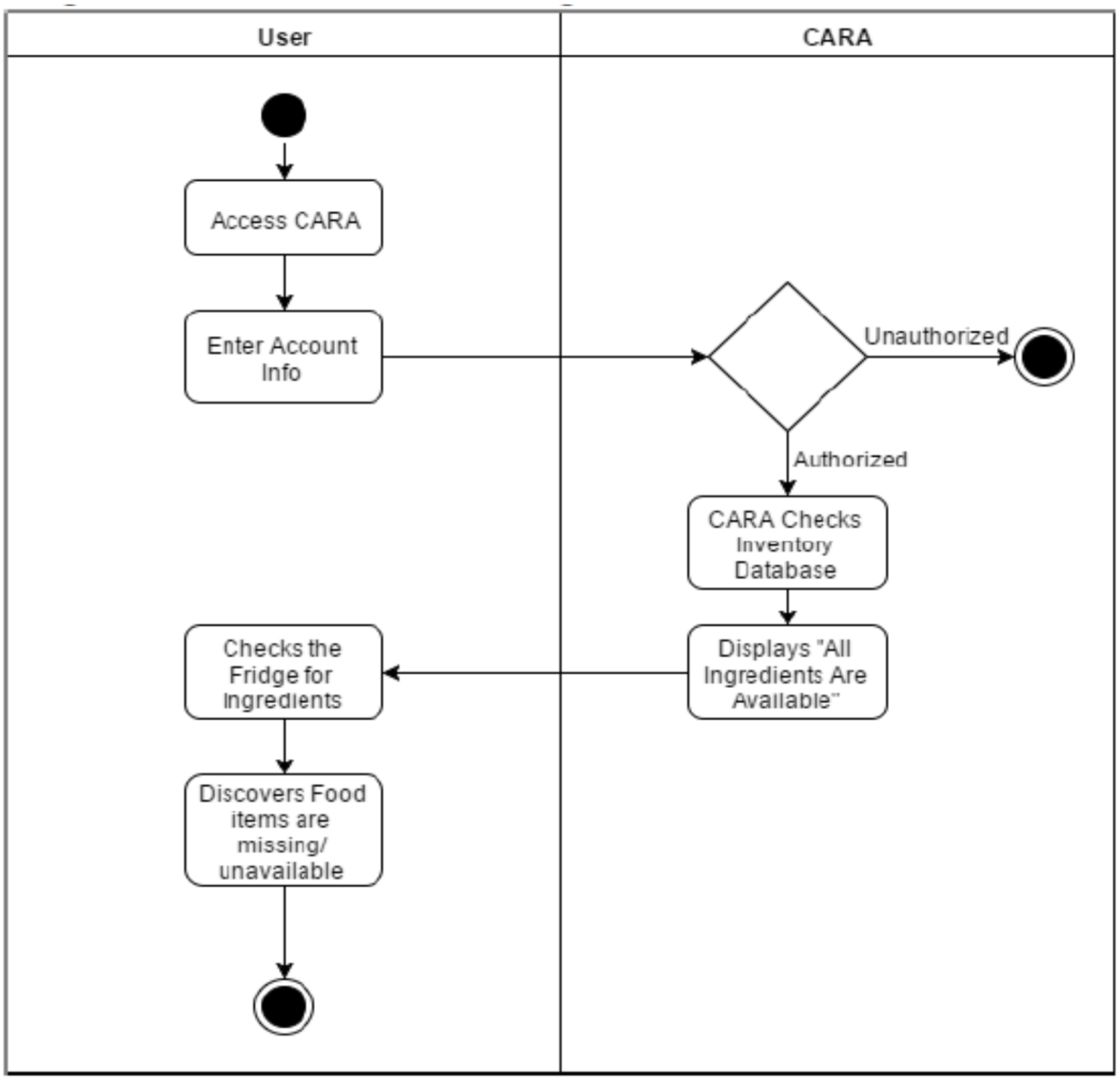
Q7 [5+5+5=15]

Activity Diagrams

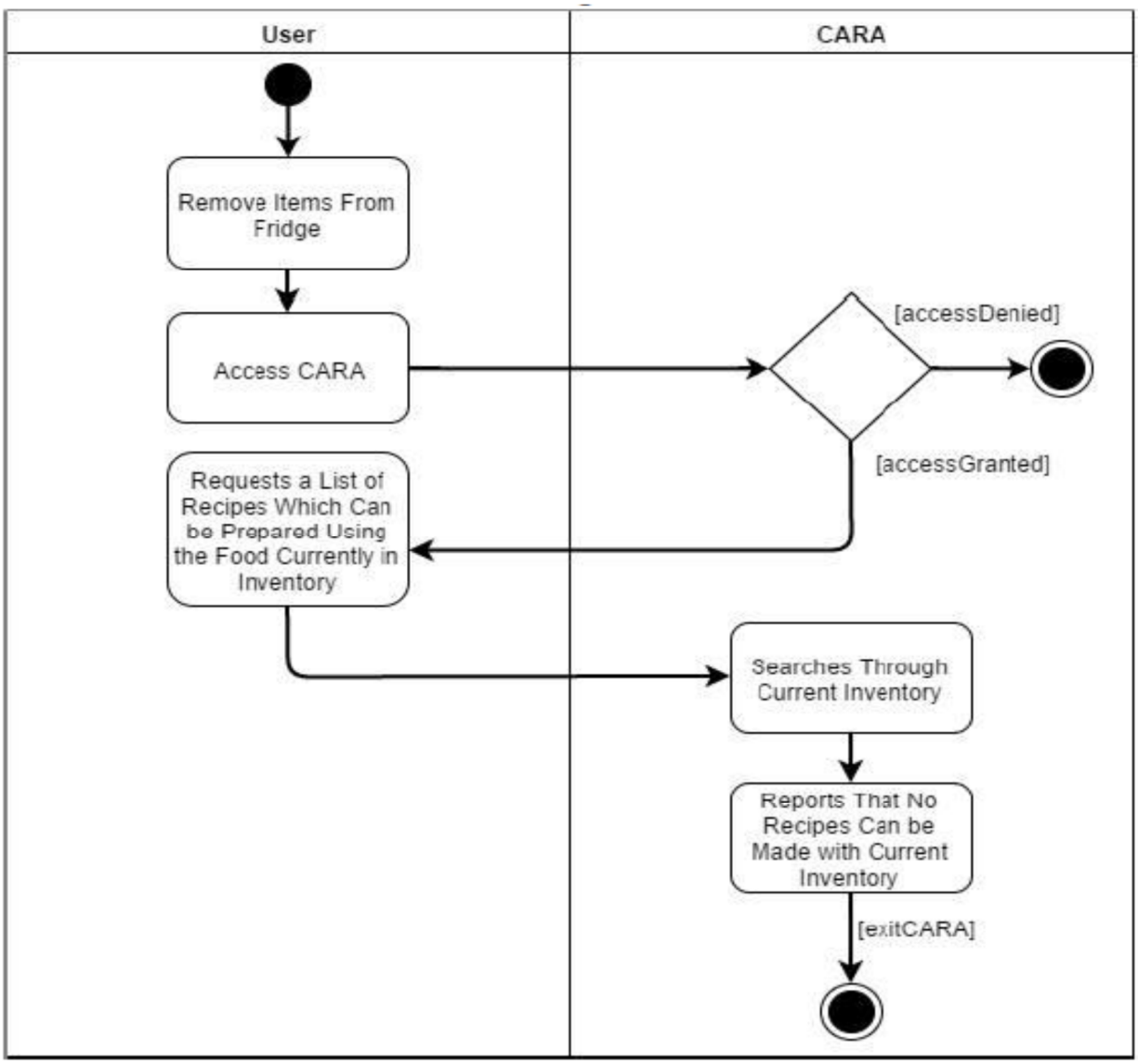
7. a) Positive



b) Negative


User
CARA
Access CARA
Enter Account Info
Unauthorized
Authorized
CARA Checks Inventory Database
Displays "All Ingredients Are Available"
Checks the Fridge for Ingredients
Discovers Food items are missing/ unavailable

c) Abnormal


User
CARA
Remove Items From Fridge
Access CARA
[accessDenied]
[accessGranted]
Requests a List of Recipes Which Can be Prepared Using the Food Currently in Inventory
Searches Through Current Inventory
Reports That No Recipes Can be Made with Current Inventory
[exitCARA]

Q8 [10=3+7]

The following is a sample solution:

1. The doors must be able to open when the elevator is fully arrived at each floor of the hotel.
Note: The additional state of available has to be added to doorState. For the door to be open it must be available (as indicated by the second statement), which only occurs when the elevator is at a floor (indicated by the first statement).

$$doorState : \{\text{closed, open, available}\}$$
$$atFloor : FLOOR \rightarrow \{true, false\}$$

$$f \in FLOORS;$$
$$e \in ELEVATORS;$$
$$b \in BUTTONS;$$

$$\forall f \in FLOOR : atFloor(f) = true \Rightarrow doorState(e) = available$$
$$doorState(e) = open \Rightarrow doorState(e) = available$$
$$\forall f \in FLOOR : atFloor(f) = false \Rightarrow doorState(e) = closed$$

2. Pressing a floor number button on the elevator will set the elevator to move in that direction to the indicated floor after the doors close.
Note: The hold direction does not need to be included because it is encompassed by the floorsPressed set being empty. There are three predicates from this requirement. The first is if a button is pressed, and that floor is above where they currently are, doors close, and the elevator begins to move up. The second is parallel to it, but if the floor is below the current floor, then the elevator goes down. The requirement doesn't state that the elevator had to be moving in that direction originally (if the elevator is going up, in practice, it should only service floors pressed that are above the current floor), so those conditions were not added. Instead, whatever is pressed, it will go to, regardless of previous direction.

$$elevatorDirection : ELEVATOR \rightarrow \{up, down, hold\}$$
$$floorsPressed : \text{set(MAX_FLOORS)}$$
$$isPressed : FLOOR \rightarrow \{true, false\}$$
$$curFloor : \rightarrow FLOOR$$

$$f \in FLOORS;$$
$$e \in ELEVATORS;$$
$$b \in BUTTONS;$$

$$(isPressed(f) \wedge (f > curFloor)) \Rightarrow$$
$$(floorsPressed' = floorsPressed \cup \{f\} \wedge doorState(e) = closed \wedge elevatorMoving = true \wedge elevatorDirection = up)$$

$$(isPressed(f) \wedge (f < curFloor)) \Rightarrow$$
$$(floorsPressed' = floorsPressed \cup \{f\} \wedge doorState(e) = closed \wedge elevatorMoving = true \wedge elevatorDirection = down)$$

3. In the case of any hotel wide emergency system activating (eg fire alarm) all elevators go to "out of service" status.

$$isEmergency : HOTELSTATE \rightarrow \{emergency, safe\}$$

$$f \in FLOORS;$$
$$e \in ELEVATORS;$$
$$b \in BUTTONS;$$

$$isEmergency(h) = emergency \Rightarrow \forall e \in ELEVATOR : elevatorStatus(e) = outService$$

Q9 [10]

A safety concern with the Cyberfridge that I believe is important to note, is possibilty of having it losing power resulting in the food inside it to spoil. This ay lead to a user to unknowingly consuming bad food and potentially falling ill. To prevent this from happening an alert can be displayed to the user on the interface of the fridge noting that food may have spoilied in the Cyberfridge when it is turned back on.

Safety Concern: Power shuts down in fridge spoiling the food.

Formal Specification:

Domain and Notation
STATE: set of all possible states of the fridge circuit power
powerOn: {true, false}
alert: {true, false}
alertAcknowledged: {true, false}
alertDisplayState: STATE {alertDisplayed, alertNotDisplayed}
e ϵ STATE

Statement
powerOn = false ➡ alert = true
alert = false alertDisplayState = alertNotDisplayed
alert = true ^ alertAcknowledged = false ^ alertDisplayState = notDisplayed ➡
alertDisplayState = alertDisplayed
alert = true ^ alertAcknowledged = false ^ alertDisplayState = alertDisplayed ➡
alertDisplayState = alertDisplayed
alert = true ^ alertAcknowledged = true ^ alertDisplayState = alertDisplayed ➡
alert = false ^ alertDisplayState = alertNotDisplayed

Q10 [6=2+2+2]

The following is a sample solution:

Express the following statements in Linear Temporal Logic:

a. *A passenger entering the elevator at 5th floor and pushing 2nd floor button, will always reach 2nd floor, even if she/he entered an upwards travelling elevator.*

In this case you might use the following atomic predicates: *floor=2, direction=up, direction=down, ButtonPres2, floor=5, etc.*

Solution: □ ((floor=5 ∧ ButtonPressed2 ∧ (direction=up ∨ direction=down)) ⇒ floor=2)

b. *It is impossible to get a state where* **started** *holds but* **ready** *does not hold.*

'*started*' and '*ready*' are atomic predicates.

Solution: □ ¬(started ∧ ¬ready)

c. *For any state, if a* **request** *(of some resource) occurs, then it will eventually be* **acknowledged**.

'*request*' and '*acknowledged*' are atomic predicates.

Solution: □ (requested ⇒ ◇ acknowledged)

Q11 [10]

The following is a sample solution:

SRC tables are most useful when observing how the CARA system works and interacts with the internet. Explicitly stating each event, condition, and transition helps to understand what how the system works and what is required of it.

Transition Table for Accessing CARA

Current State	Event	New State
login	@T (accountInfo = true)	systemUI
ogin	@T (accountInfo = false)	login
systemUI	@T (requestInventory = true)	output
systemUI	@T (modifyInventory = true)	updateSystem
output	@T (displayedInv = true)	systemOff
updateSystem	@T (furtherRequests = false)	systemOff

Event Table for the term Connected

State	Events	
login sytemUI output updateSystem	@T (serverAccess = off)	@T (reconnect = true) WHEN serverAccess = off
Connected	False	True

Condition Table for Variable

State	Conditions	
systemOff	**NOT powerOff AND NOT** accountInfo = flase	powerOff **OR** accountInfo = false
systemUI output updateSystem	True	False
Access CARA	Authorized	notAuthorized

SE 3RA3 Assignment 2 Question 1(a)

IEEE Std-830 Requirements Specifications Documents

Table of Contents

1 Introduction 2
1.1 Purpose 2
1.2 Scope 2
1.3 Definitions, Acronyms, Abbreviations 3
1.4 References 3
1.5 Overview 3

2 General Description 4
2.1 Product Perspective 4
2.2 Functions 4
2.3 User Characteristics 4
2.4 General Constraints 5
2.5 Assumptions & Dependencies 5
2.6 Apportioning of Requirements 5

3 Specification Requirements 5
3.1 Functional Requirements 5
3.2 External Interface Requirements 6
3.3 Performance Requirements 6
3.4 Design Constraints 6
3.5 Software Quality Attributes 6
3.6 Other Requirements 6

1 Introduction

The CyberFridge project aims to improve and add further functionality to a common household appliance. The CyberFridge uses Internet connectivity, vision and mechanical systems to create an intelligent and productive refrigerator. Most specifically, its software, the CARA system is being implemented to provide various services which will improve the consumer's day to day life and interactions with their fridge.

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to provide a complete description of the various requirements which are needed to design the CARA system used in the CyberFridge. This document is meant to further develop our understanding of the software system and all of its requirements. This document is intended for the use of all the stakeholders participating in the CyberFridge project, specifically those involved in the development of the CARA software system. These stakeholders include the various consumers such as business owners, home owners, house keepers, restaurant owners, and chef/cooks. Other stakeholders which are responsible for the development of the software include the project manager/leader, engineers, programmers, the architect, the security analyst, and possible replacement personnel.

1.2 Scope

The CARA system and CyberFridge is meant to be deployed in a household or business setting, or anywhere a refrigerator is used. The CARA system which is to be specified in this document handles all of the software aspects to the CyberFridge and though it makes use of the fridge's hardware, it does not handle the physical mechanisms. The following document entails the specification of the software only. The goal of creating this software is to create a fridge system which makes everyday tasks relating to the use of a refrigerator simpler, more convenient, and more efficient.

1.3 Definitions, Acronyms, Abbreviations

This section lists key terms/abbreviations which may be used throughout the document.
Acronyms and Abbreviations Table

Symbol	Description
SRS	Software Requirements Specification
GUI	Graphic User Interface
UI	User Interface
OS	Operating System
CARA	Clifford Automated Recipe Agent

List of Definitions:

- CARA - this refers to the software portion of the CyberFridge and will serve the purpose of electronically storing recipes, keeping track of the fridge inventory, etc.

1.4 References

This document references the IEEE Std-803 template for Requirements Specification which can be found in SE 3RA3 Lecture notes 12. The Cyber-Fridge and CARA system description are referenced from Assignment 1.

1.5 Overview

The remainder of the SRS will outline the general description of the software, any constraints, assumptions and dependencies, and most importantly and specific requirements of the CARA system which will be described in great detail. The Specific Requirements section of this document will go over the functional requirements and the various non-functional requirements.

2 General Description

2.1 Product Perspective

The CARA software system which specified in this SRS is a part of a larger system, they CyberFridge system. Hence, the software must be able to interface with the fridge system's hardware/ physical mechanisms such as the display on the fridge, the sensors used to track inventory and any buttons on the fridge itself.

2.2 Functions

This subsection will describe the main functions CARA must have.

- Keep Track of Inventory: the software works with the sensors (hardware device) to maintain a record of what food ingredients are in the fridge.

- Online Access: ability to access CARA online and edit/modify the recipes while also being able to check ingredients present in the fridge.
- Maintains Database: the software systems keeps track of all the information stored in the database and creates backups in case of system failure.

2.3 User Characteristics

Some assumptions about the user includes:

- The user does not require any experience with the CyberFridge or CARA.
- The user must be able to read/understand the language CARA is (i.e. English).
- The user must have electricity to power the CyberFridge.

2.4 General Constraints

The constraints for this project will mainly be budget and time constraints. The cost of the software must be no more than the budget allocated by the project manager so as to earn a profit when distributing the system to clients. The time constraints present would be to meet all the deadlines/milestone dates and essentially complete the CARA system in time for the release of the CyberFridge.

2.4 Assumptions & Dependencies

Environment assumptions for our system include:

- Availability of internet connection at all times.
- Availability of electricity to properly power the system at all times.
- Proper hardware to make use of the CARA software.

2.5 Apportioning of Requirements

Requirements which can be delayed until future versions of the system are released would include the need for software updates, hardware and software maintenance, aesthetics and user friendly UI/GUI which is not a main priority of the system.

3 Specific Requirements

3.1 Functional Requirements

Functional Requirements include:

- The system shall allow users to use a recipe system to determine what recipe they want to _x.
- The system shall maintain a database of various recipes.
- The recipe agent shall electronically create a shopping list to inform users of items which are lacking for specific recipes while also informing the user of what is present in the CyberFridge.
- The system should be able to reference what is in the fridge and compare it to the recipe database.
- The database should be able to record and sort the various recipes in different categories.
- The system must maintain a counter to record the most popular dishes previously selected in relation to other dishes.
- Users should be able to add new recipes to the system.
- CARA food tracker must keep track of the amount of items in stock and their expiration dates.
- The system must be able to output a list of all food items in inventory.
- The user shall be able to set the criteria in which the food list is out putted (i.e. food categories, alphabetical list, time remaining before expire, etc.)
- The system should be able to return information on any item requested by the user (i.e. quantity on hand and the time remaining before it must be removed from inventory)
- The system must update the database when users add items or remove them from inventory.
- Users must be able to create or modify categories in the system while also being able to modify the recipes and where they belong.
- Users must be able to download recipes remotely from outside servers through the use of internet

- The system shall be able to output a list of possible recipes which can be prepared using the food on hand.
- The system shall be able to output a list of food items needed to be purchased in order to prepare a particular recipe or group of recipes.
- The system should be able to output a list of suggested recipes based on criteria set by the user (i.e. recipes for special occasions, recipes using or not suing specific food items and recipes which utilize food which will need to be disposed of in the near future due to expire dates)
- System must be able to output lists of recipes based on user's favorite recipes, recipes not used recently or recipes ordered by preparation time.
- Users should be able to access CARA through the internet.
- The system must have an internet-accessible interface which will allow users to access capabilities of the CyberFridge through the World Wide Web.

3.2 External Interface Requirements

- The CARA system must be able to communicate with the CyberFridge through the use of the internet.
- The software must send and receive signals from the product scanner which keeps track of the inventory of the fridge.
- The software must respond to signals sent by buttons and other hardware components of the fridge.
- The software shall interact with local power systems to obtain electricity which will essentially power the whole CyberFridge.
- The software is expected to be able to communicate with devices of varying OS such as Windows, Apple, or Linux.

3.3 Performance Requirements

The following is a list of performance requirements the software must meet:

- Speed Requirement: the system shall be able to load under 30 sec on a Bell Sympatico or equivalent connection.
- Capacity Requirement: the database must come with 13 recipes divided into 3 categories which will be picked from cookbooks or for the web as needed.
- Reliability and availability Requirement: The product should be available for use 24 hours per day, 365 days per year.
- The internet-accessible interface should also be accessible at all times.

3.4 Design Constraints

The following is a list of development requirements the software must meet:

- The system must be programmed using Java. The system must be cross-platform and work on Windows, Apple and Linux devices.
- The system must be linked to a powerful server to store its database. The development of the software must be completed within the assigned time frame and budget.

3.5 Software Quality Attributes

The following is a list of security, maintainability and portability requirements:

- Users must have a password to be allowed access to the recipe-agent system.
- A client side refresh is required to trigger any database accesses or display events.
- The system needs to keep multiple copies of all necessary data and code being saved to prevent data loss in the case of a system crash.
- The system should run on Google Chrome/Explorer/Firefox running on PC under Windows 7 or Apple under Safari or Windows 7 + Parallel.

3.6 Other Requirements

The following is a list of other requirements such as Usability, Operational and look and feel:

- Ease of use: the CyberFridge and CARA should be easy to use by a wide range of people (i.e. from ages 12 and up) as the system is meant to be implemented in a common household setting.
- Ease of learning: it should be easy to learn how to use the product as it may be used in a variety of different settings such as in homes, businesses, restaurants, etc.
- A TCP/IP connection is required to access the Internet.
- Programmed in Java so it is platform-independent and should run on any target platform without modification but is specifically made to run and is tested on a Windows 7 platform.
- Expected physical environment: The product is meant to be used by the consumer in a kitchen while also being able to access the CyberFridge capabilities via internet.
- The CyberFridge should have a display which users can use to view information outputted by CARA.

SE 3RA3 Assignment 2 Question 1(b)

Volere Requirements Specification Documents

Table of Contents

1 Project Drivers 2
1.1 Purpose of the Project 2
1.2 The Client, the Customer and Other Stakeholders 2
1.3 Users of the Product 2

2 Project Constraints **3**
2.1 Mandated Constraints 3
2.2 Naming Conventions and Definitions 3
2.3 Relevant Facts and Assumptions 3

3 Functional Requirements 4
3.1 The Scope of the Work 4
3.2 The Scope of the Product 4
3.3 Functional and Data Requirements 4

4 Nonfunctional Requirements 5
4.1 Look and Feel Requirements 5
4.2 Usability and Humanity Requirements 5
4.3 Performance Requirements 5
4.4 Operational and Environmental Requirements 5
4.5 Maintainability and Support Requirements 6
4.6 Security Requirements 6
4.7 Cultural and Political Requirements 6
4.8 Legal Requirements 6

5 Project Issues 6
5.1 Open Issues 6
5.2 O_-the-Shelf Solutions 7
5.3 New Problems 7
5.4 Tasks 7
5.5 Migration to the New Product 7
5.6 Risks 7
5.7 Costs 7
5.8 User Documentation and Training 7
5.9 Waiting Room 7
5.10 Ideas for Solutions 7

1 Project Drivers

1.1 Purpose of the Project

The purpose of this Software Requirements Specification (SRS) document is to provide a complete description of the various requirements which are needed to design the CARA system used in the CyberFridge. This document is meant to further develop our understanding of the software system and all of its requirements.

1.2 The Client, the Customer and Other Stakeholders

This document is intended for the use of all the stakeholders participating in the CyberFridge project, specifically those involved in the development of the CARA software system. These stakeholders include the various consumers such as business owners, home owners, house keepers, restaurant owners, and chef/cooks. Other stakeholders which are responsible for the development of the software include the project manager/leader, engineers, programmers, the architect, the security analyst, and possible replacement personnel.

1.3 Users of the Product

- General Public (Primarily Home Owners or Chefs) - The users shall access CARA through the use of the buttons and display on the CyberFridge or online through the internet-accessible interface.
- Maintenance Users and Service Technicians - The software will be updated and adjusted consistently, the professionals working on these updates will require the product to have some features to facilitate modifications.

2 Project Constraints

2.1 Mandated Constraints

Requirements which can be delayed until future versions of the system are released would include the need for software updates, hardware and software maintenance, aesthetics and user friendly UI/GUI which is not a main priority of the system.

2.2 Naming Conventions and Definitions

This section lists key terms/abbreviations which may be used throughout the document.

Acronyms and Abbreviations Table

Symbol	Description
SRS	Software Requirements Specification
GUI	Graphic User Interface
UI	User Interface
OS	Operating System
CARA	Clifford Automated Recipe Agent

List of Definitions:

- CARA - this refers to the software portion of the CyberFridge and will serve the purpose of electronically storing recipes, keeping track of the fridge inventory, etc.

2.3 Relevant Facts and Assumptions

Relevant facts and assumptions for our system include:

- Availability of internet connection at all times.
- Availability of electricity to properly power the system at all times.
- Proper hardware to make use of the CARA software.

3 Functional Requirements

3.1 The Scope of the Work

The scope of our work focuses on the CARA software which is the main focus of the project but will also interact with the hardware components. Work on CARA will focus on its design and implementation to meet all of the requirements and constraints.

3.2 The Scope of the Product

The CARA system and CyberFridge is meant to be deployed in a house hold or business setting, or anywhere a refrigerator is used. The CARA system which is to be specified in this document handles all of the software aspects to the CyberFridge and though it makes use of the fridge's hardware, it does not handle the physical mechanisms. The following document entails the specification of the software only. The goal of creating this software is to create a fridge system which makes everyday tasks relating to the use of a refrigerator simpler, more convenient, and more efficient.

3.3 Functional and Data Requirements

Functional Requirements include:

- The system shall allow users to use a recipe system to determine what recipe they want to _x.
- The system shall maintain a database of various recipes.
- The recipe agent shall electronically create a shopping list to inform users of items which are lacking for specific recipes while also informing the user of what is present in the CyberFridge.
- The system should be able to reference what is in the fridge and compare it to the recipe database.
- The database should be able to record and sort the various recipes in different categories.
- The system must maintain a counter to record the most popular dishes previously selected in relation to other dishes.
- Users should be able to add new recipes to the system.
- CARA food tracker must keep track of the amount of items in stock and their expiration dates. The system must be able to output a list of all food items in inventory. The user shall be able to

set the criteria in which the food list is out putted (i.e. food categories, alphabetical list, time remaining before expire, etc.)

- The system should be able to return information on any item requested by the user (i.e. quantity on hand and the time remaining before it must be removed from inventory)
- The system must update the database when users add items or remove them from inventory.
- Users must be able to create or modify categories in the system while also being able to modify the recipes and where they belong.
- Users must be able to download recipes remotely from outside servers through the use of internet
- The system shall be able to output a list of possible recipes which can be prepared using the food on hand.
- The system shall be able to output a list of food items needed to be purchased in order to prepare a particular recipe or group of recipes.
- The system should be able to output a list of suggested recipes based on criteria set by the user (i.e. recipes for special occasions, recipes using or not suing specific food items and recipes which utilize food which will need to be disposed of in the near future due to expire dates)
- System must be able to output lists of recipes based on user's favorite recipes, recipes not used recently or recipes ordered by preparation time.
- Users should be able to access CARA through the internet.
- The system must have an internet-accessible interface which will allow users to access capabilities of the CyberFridge through the World Wide Web.

4 Nonfunctional Requirements

4.1 Look and Feel Requirements

- The CyberFridge should have a display which users can use to view information outputted by CARA.

4.2 Usability and Humanity Requirements

- Ease of use: the CyberFridge and CARA should be easy to use by a wide range of people (i.e. from ages 12 and up) as the system is meant to be implemented in a common household setting.
- Ease of learning: it should be easy to learn how to use the product as it may be used in a variety of different settings such as in homes, businesses, restaurants, etc.

4.3 Performance Requirements

- Speed Requirement: the system shall be able to load under 30 sec on a Bell Sympatico or equivalent connection.
- Capacity Requirement: the database must come with 13 recipes divided into 3 categories which will be picked from cookbooks or o_ the web as needed.
- Reliability and availability Requirement: The product should be avail- able for use 24 hours per day, 365 days per year.
- The internet-accessible interface should also be accessible at all times.

4.4 Operational and Environmental Requirements

- A TCP/IP connection is required to access the Internet. Programmed in Java so it is platform-independent and should run on any target platform without modification but is specifically made to run and is tested on a Windows 7 platform.

- Expected physical environment: The product is meant to be used by the consumer in a kitchen while also being able to access the CyberFridge capabilities via internet.

4.5 Maintainability and Support Requirements

- The system should run on Google Chrome/Explorer/Firefox running on PC under Windows 7 or Apple under Safari or Windows 7 + Parallel

4.6 Security Requirements

- Users must have a password to be allowed access to the recipe-agent system.
- A client side refresh is required to trigger any database accesses or display events.
- The system needs to keep multiple copies of all necessary data and code being saved to prevent data loss in the case of a system crash.

4.7 Cultural and Political Requirements

- N/A

4.8 Legal Requirements

- N/A

5 Project Issues

5.1 Open Issues

- N/A - no issues raised at the time, the CyberFridge system is still in the development phase.

5.2 Off-the-Shelf Solutions

Ready-Made Products:

- N/A - no other products like the CyberFridge yet. Products That Can Be Copied:
- Similar products which we can reference would be online recipe books/shopping lists as CARA is basically an online recipe agent for the CyberFridge system.

5.3 New Problems

Effects on the Current Environment:

- The CyberFridge system will make interactions with a refrigerator simple, efficient and more convenient. The system will help users keep track of what they have in the fridge and also help

them decided on what they need to get from stores based on their current inventory or what recipe they wish to make. By using CARA, it will reduce the time it takes to complete these tasks. Potential User Problems:

- The user must adapt and learn how to use the system, along with its interface.

5.4 Tasks

Project Planning:

- The software system to be should last as long as the CyberFridge and should have scheduled maintenance and updates on each device to ensure all users able to use the product to its fullest. Planning of the Development Phases:
- A detailed project timeline is necessary to keep workers on track and ensure specific milestone dates are met. It would also be a good idea to break up tasks into different phases so as to take on the project in a step by step process.

5.5 Migration to the New Product

Requirements for Migration to the New Product:

- The implementation of the new system is to be broken down into phases which would focus on different aspects of CARA. Phases would include ensuring the system keeps track of food items, the system keeps track of recipes, and the user is able to access CARA through the internet.
- A backup for the new system will be implemented so as to prevent the loss of data that is stored on the servers.

5.6 Risks

The risks present in the system include:

- Loss of Recipes
- Loss of Privacy
- Unable to Access System
- Inaccurate Database
- Corrupted Data
- Inaccurate Food Item Information

5.7 Costs

The cost for this project must be within the project budget assigned by the project manager. Furthermore, keeping the costs to produce the product lower than the selling price would provide more profit and revenue to the company.

5.8 User Documentation and Training

The product must meet the ease of use and ease of learning requirements, thus the required documentation to be supplied to the user is very minimal. A short user manual and setup document would suffice in helping users understand how to use the system.

5.9 Waiting Room

Similar to mandated constraints, requirements which can be delayed until future versions of the system are released would include the need for software updates, hardware and software maintenance, aesthetics and user friendly UI/GUI which is not a main priority of the system. Furthermore, other requirements may appear throughout the software development process and can be addressed later on.

5.10 Ideas for Solutions

- N/A